AMENDMENTS TO THE CLAIMS:

This listing of claims will replace all prior versions, and listings, of claims in the application:

LISTING OF CLAIMS:

1. (Currently Amended) A method of providing a Certificate Status Service ("CSS") for checking validities of authentication certificates issued by respective issuing Certification Authorities ("CAs"), comprising the steps of:

receiving one or more certificate status queries from requesting entities;

if the issuing CAs are not found on a CSS's list of approved CAs or the certificates have expired, returning invalid statuses for those certificates;

if the current statuses are found in the CSS's status cache, returning those certificates' statuses;

if any status has not yet been determined, ~~identifying~~ fetching all certificate status reporting methods and communications information from a configuration store of the CSS that are needed for retrieving a status of each ~~an authentication~~ certificate whose status has not yet been determined from [[an]] the respective issuing CAs ~~CA that issued the authentication certificate~~;

configuring [[a]] ~~connector~~ connectors based on the identified information for communicating with the issuing [[CA]] CAs;

communicating with the issuing [[CA]] CAs according to the configured ~~connector when the status of the authentication certificate is queried; and~~ connectors;

retrieving the status of [[the]] all queried ~~authentication certificate~~ certificates;

processing the certificate statuses according to an appropriate certificate status reporting method that may include, but is not limited to, Certificate Revocation Lists (CRLs) that are retrieved at specified publication intervals, Delta Certificate Revocation Lists (ΔCRLs) that are retrieved upon notification, LDAP, OCSP, and any other certificate status means that are queried and retrieved using real-time protocols;

recording retrieved certificate statuses in the CSS's cache memory;

returning the retrieved certificate statuses to the requesting entities;

wherein the issuing [[CA]] CAs and [[the]] connector parameters are designated on a list of approved CAs in [[a]] the configuration store that enable the CSS to interwork with any CAs and CA domains even though they can operate using dissimilar certificate practices and policies.

2. (Currently Amended) The method of claim 1, wherein a certificate is deemed to have expired if a local date and time ~~are checked for whether they~~ fall ~~within a~~ outside a validity period as indicated in the ~~authentication~~ certificate ~~and an invalid status is reported if the local date and time fall outside the validity period~~.

3. (Currently Amended) The method of claim [[1]] 2, wherein the issuing CA is ~~included in the~~ added to at least one organization's list of approved CAs by vetting and approving the issuing CA according to predetermined business rules, wherein the business rules include at least one rule for reviewing the acceptability of the CA's certificate policy and practices for insuring the identity of the entity requesting the certificate, and if the issuing CA is vetted and not approved or later disapproved, the issuing CA is ~~designated on a~~ added to the organization's list of not-approved CAs in the configuration store and/or has any prior entry removed from the organization's list of approved CAs.

4. (Currently Amended) The method of claim 3, wherein vetting and approving the issuing CA ~~includes~~ include registering a representation of a ~~the CA's~~ trusted authentication certificate of the CA with the CSS and adding ~~at least the representation,~~ at least a status reporting component of the CA, the certificate status reporting method including, but not limited to, CRL, OCSP, LDAP, ~~and~~ a time-to-live data element, and communication information needed to configure a connector to the CSS's configuration store. ~~to a local cache memory, and a connector is configured for retrieving the added status when the status of the trusted authentication certificate is queried.~~

5. (Currently Amended) The method of claim [[2]] 4, further comprising the steps of:

checking and updating a local cache memory for the certificate status, and if the status is found in the local cache memory [[and]], checking that the local date and time are within the certificate's validity period~~, retrieving the status from the local cache memory, or if~~ and that the time-to-live data element [[or]] and use-counter values are within a threshold;

if any of the validity period, time-to-live data element, or use-counter values are unacceptable, ~~is exceeded~~ clearing the local cache memory ~~entry~~, wherein if the status is not found in the local cache memory, the CSS establishes a communication session with [[a]] the certificate status reporting component of the issuing CA, composes a certificate status request using one of the CRL or real-time reporting methods according to the configured connector, retrieves the status from the certificate status reporting component, closes the communication session with the certificate status reporting component, and adds at least one of the ~~authentication~~ certificate's identification, status, use-counter, and time-to-live data element to the local cache memory.

6. (Currently Amended) The method of claim 1, wherein the certificate status reporting method is indicated [[by]] to be a CRL ~~Certificate Revocation List (CRL)~~, according to a publication schedule of the issuing CA, wherein the CSS retrieves the CRL from a certificate status reporting component listed in the configuration store, the CSS clears [[a]] the cache memory associated with the issuing CA, and the CSS ~~determines~~ extracts the status of [[the]] all authentication ~~certificate~~ certificates from the CRL and

stores the ~~status~~ statuses in the cache memory associated with the issuing CA.

7. (Currently Amended) The method of claim 1, wherein the certificate status reporting method is indicated [[by]] to be a ΔCRL ~~Delta Certificate Revocation List ("ΔCRL");~~ wherein upon notification by the issuing CA that [[a]] the ΔCRL is available, the CSS retrieves the ΔCRL from a certificate status reporting component listed in the configuration store[[;]] and if the ΔCRL is a ~~complete~~ full CRL, then the CSS clears [[a]] the cache memory associated with the issuing CA, ~~determines the status~~ extracts all certificate statuses from the CRL, and stores the ~~status~~ statuses in the cache memory[[;]], and if the ΔCRL contains ~~only~~ changes occurring after publication of a full CRL, the CSS ~~determines the status~~ extracts all certificate statuses from the ΔCRL, and stores the ~~status~~ statuses in the cache memory.

8. (Currently Amended) The method of claim 1, wherein the communicating step includes communicating according to a ~~sequence~~ plurality of connectors to multiple CAs and PKIs.

9. (Currently Amended) The method of claim 1, wherein [[a]] the connector ~~embeds~~ allows more than one certificate status ~~check~~ request to be chained together in a single communicating step.

10. (Currently Amended) The method of claim 1, wherein the ~~authentication certificate~~ certificates are [[is]] held in the configuration store until expiration and information are extracted as needed ~~not used for identification~~.

11. (Currently Amended) [[A]] The method of claim 1, wherein the retrieving [[a]] of the status of ~~an authentication~~ the certificate issued by [[an]] the ~~issuing~~ approved CA ~~Certification Authority ("CA")~~ in response to a query from a trusted third-party repository of information objects to the CSS [[a]] ~~Certificate Status Service ("CSS")~~ to validate the authentication certificate's status~~, comprising~~ comprises the steps of:

locating and reporting the status if the status is present and current in [[a]] the cache memory of the CSS;

~~otherwise~~ if the status is not present in the cache memory, performing the steps of:

obtaining [[a]] the communications information, status type, and retrieval method from [[a]] the CSS configuration store;

if the status type is CRL ~~Certificate Revocation List ("CRL")~~ and the CRL in the cache memory is current, and ~~the last retrieved CRL is current,~~ [[but]] the status is not found in the cache memory, then reporting the status as valid;

if the CRL is not current or found in the cache memory ~~if the status type is not CRL~~ and local time is greater than a next scheduled

publication time for the CRL or the status type is not CRL, creating a connector and [[then]] composing a certificate status request according to the status type;

establishing a communication session with ~~the issuing CA~~ a status reporting component of the issuing CA;

retrieving the status from [[a]] the CA's status reporting component ~~of the issuing CA~~ using the obtained retrieval method and ending the communication session;

interpreting the retrieved status;

associating, with the interpreted retrieved status, a time-to-live value representing a period specified by [[a]] the respective CSS policy for the status type;

adding at least one of the ~~authentication~~ certificate's identification, status[[,]] and time-to-live values to the cache memory; and

reporting the status to the trusted third-party repository of information objects ~~in response to the query~~.

12. (Cancelled)

13. (Cancelled)

14. (Cancelled)

15. (Currently Amended) [[A]] The ~~Certificate Status Service~~ [[("CSS")]] method of claim 1 for providing ~~accurate and timely~~ certificate status ~~indications of~~ reports for authentication certificates issued by the approved CAs ~~issuing Certification Authorities ("CAs")~~, further comprising:

~~providing a~~ reporting valid certificate status when the status type is CRL, the CRL is current, ~~status of an authentication certificate as~~ and the status is not found in the cache memory; ~~indicated by a Certificate Revocation List ("CRL") when the certificate's issuing CA uses CRLs for indicating status;~~

~~otherwise, providing the~~ reporting the status when status is found in the ~~indicated by a~~ cache memory ~~when the cache memory includes a status and a~~ and the time-to-live and use-counter ~~data element~~ values have [[is]] not exceeded respective thresholds;

if either the time-to-live or use-counter ~~data element~~ values have exceeded the threshold ~~is exceeded,~~ clearing the status from the cache memory;

if the certificate status has not been reported in a previous step, then requesting and retrieving the status using the status type indicated in the configuration store;

when the status type is CRL, retrieving and parsing the new CRL at a next indicated publication time;

when the status type is at least one of the type LDAP, OSCP, and any other [[a]] real-time certificate status reporting protocol, retrieving and parsing ~~when~~ the status; ~~is not in the cache memory;~~

adding at least one of the certificate's identification, status, [[and]] time-to-live and use-counter ~~data element~~ values to the cache memory; and

~~providing~~ reporting the retrieved status to the requesting entity.

16. (Currently Amended) The CSS of claim 15, wherein a status use-counter data element is added to the cache memory, wherein[[;]] the status use-counter data element is incremented or decremented every time the certificate's status is checked,[[;]] and if the status use-counter data element passes a threshold, then the status is ~~provided~~ reported and the cache memory is cleared with respect to the status.

17. (Currently Amended) The CSS of claim 16, wherein a status last-accessed data element is added to the cache memory, and the status last-accessed data element in conjunction with the status use-counter data element enable the CSS to determine an ~~determination of an~~ activity level of the certificate's status.

18. (Currently Amended) The CSS of claim 17, wherein when a request is made to the CSS to retrieve a status of a new certificate and the cache memory has reached an allocated ~~buffer~~ memory size limit, the CSS searches the cache memory for [[a]] every certificate status entry where the current time exceeds the time-to-live ~~data element~~ value, for every certificate status entry where the value of the use-counter data element exceeds the threshold and the value of the at least one certificate status entry with the

oldest ~~exceeds the current local time or a~~ last-accessed value, wherein ~~data element indicating an oldest date and~~ the CSS then clears the respective cache memory ~~entry;~~ entries, ~~and the CSS then~~ retrieves the requested certificate status, places [[it]] the certificate status in the cache memory, and ~~provides~~ reports the requested certificate status to the requesting entity.

19. (Withdrawn) A method of executing a transaction between a first party and a second party by transferring control of an authenticated information object having a verifiable evidence trail, comprising the steps of:

retrieving an authenticated information object from a trusted third-party repository of information objects, wherein the authenticated information object includes a first digital signature block comprising a digital signature of a submitting party and a first authentication certificate relating at least an identity and a cryptographic key to the submitting party, a date and time indicator, and a second digital signature block comprising a second digital signature of the trusted third-party repository of information objects and a second authentication certificate relating at least an identity and a cryptographic key to the trusted third-party repository of information objects; the first digital signature block was validated by the trusted third-party repository of information objects; and the authenticated information object is stored as an authoritative copy information object under the control of the trusted third-party repository of information objects;

executing the retrieved authenticated information object by the second party by including in the retrieved authenticated information object a third

digital signature block comprising at least a third digital signature and a third authentication certificate of the second party; and

forwarding the executed retrieved authenticated information object to a trusted third-party repository of information objects, wherein the trusted third-party repository of information objects verifies digital signatures and validates authentication certificates associated with the digital signatures included in information objects by at least retrieving status of the authentication certificates from a Certificate Status Service ("CSS") provided according to claim 1; the trusted third-party repository of information objects rejects a digital signature block if the respective digital signature is not verified or the status of the respective authentication certificate is expired or is revoked; and if at least one signature block in the information object is not rejected, the trusted third-party repository of information objects appends the trusted third-party repository's digital signature block and a date and time indicator to the information object and takes control of the object on behalf of the first party.

20. (Withdrawn) The method of claim 19, wherein a signature block includes at least one hash of at least a portion of the information object in which the signature block is included, the at least one hash is encrypted by the cryptographic key of the block's respective signer, thereby forming the signer's digital signature, and the signer's digital signature is included in the signature block with the signer's authentication certificate.

21. (Withdrawn) The method of claim 20, wherein the executing step includes displaying a local date and time to the second party, affirming, by the second party, that the displayed local date and time are correct, and correcting the local date and time if either is incorrect.

22. (Withdrawn) The method of claim 19, wherein if the trusted third-party repository of information objects rejects a digital signature block, the trusted third-party repository of information objects requests a remedy that requires the digital signature to be recomputed and the signature block to be reforwarded.

23. (Withdrawn) The method of claim 19, wherein the trusted third-party repository of information objects checks the local date and time for accuracy and that they are within a validity period indicated by the second party's authentication certificate.

24. (Withdrawn) The method of claim 23, wherein if the local date and time are not within the validity period indicated by the second party's authentication certificate, the trusted third-party repository of information objects notifies the second party that the authentication certificate is rejected and the first party that the transaction is incomplete.

25. (Withdrawn) The method of claim 19, wherein one or more digitized handwritten signatures are included in the information object, and

placement of the digitized handwritten signatures in a data structure is specified by at least one signature tag.

26. (Withdrawn) The method of claim 19, wherein placement of one or more signature blocks in a data structure is specified by at least one signature tag.

27. (Withdrawn) The method of claim 26, wherein one or more signature blocks are separately forwarded to the trusted third-party repository of information objects with respective signature tags, and the trusted third-party repository of information objects validates the signature blocks by:

rejecting a signature block if either the respective digital signature is not verified or the respective authentication certificate is not validated, and

placing the signature block according to the respective signature tag if the signature block is not rejected,

wherein, to signature blocks sent separately, the trusted third-party repository of information objects adds a date and time indication to each signature block and appends according to business rules the trusted third-party repository's signature block in a wrapper that encompasses the information object and placed signature blocks.

28. (Withdrawn) The method of claim 27, wherein the trusted third-party repository of information objects verifies a digital signature and validates an authentication certificate in a signature block by:

determining from the business rules whether a party associated with the authentication certificate has authority,

verifying the party's digital signature,

checking that the authentication certificate's validity period overlaps the trusted third-party repository's current date and time,

checking that the local date and time falls within an allowable deviation from the trusted third-party repository 's current date and time, and

retrieving status of the authentication certificate from the CSS, and

if any of the preceding steps results in an invalid or false output, the digital signature is deemed invalid, the transaction is not executed, otherwise the digital signature is deemed valid and the transaction is executed.

29. (Withdrawn) The method of claim 19, wherein the CSS provides authentication certificate status to the trusted third-party repository of information objects by at least the steps of checking a local cache memory for the status, and if the status is found in the local cache memory and the local date and time are within the validity period, and retrieving the status from the local cache memory; or if the time-to-live or use-counter threshold is exceeded clearing the cache memory entry, wherein if the status is not found in the local cache memory, the CSS establishes a communication session with a certificate status reporting component of the issuing CA, composes a certificate status request, retrieves the status from the certificate status reporting component, closes the communication session with certificate status reporting component, and adds at least the authentication certificate's

identification, status, and a time-to-live data element to the local cache memory.

30. (Withdrawn) The method of claim 19, wherein the first party is a first trusted third-party repository of information objects and the transaction is for transferring custody of one or more authoritative copies to the first trusted third-party repository of information objects from a second trusted third-party repository of information objects, an owner of the transaction provides the second trusted third-party repository of information objects with a manifest that identifies authoritative copies to be transferred to the first trusted third-party repository of information objects, the second trusted third-party repository of information objects establishes communication with the first trusted third-party repository of information objects and identifies the purpose of its actions, the manifest is communicated to the first trusted third-party repository of information objects so that it is able to determine when the transfer of custody has been completed, the second trusted third-party repository of information objects transfers each identified authoritative copies to the first trusted third-party repository of information objects, the first trusted third-party repository of information objects retrieves status of the second trusted third-party repository's certificate and verifies the second trusted third-party repository's digital signature on each transferred authoritative copies, if any of the second trusted third-party repository's digital signatures or certificates are invalid, then the first trusted third-party repository of information objects notifies the second rusted third-party repository of

information objects and seeks a remedy, if the second trusted third-party repository of information objects does not provide a remedy, the first trusted third-party repository of information objects notifies the transaction owner that the requested transfer of custody has failed, otherwise the second trusted third-party repository of information objects creates a new wrapper for each successfully transferred information object, adding a date-time stamp and the first trusted third-party repository's signature block.

31. (Withdrawn) The method of claim 30, wherein the transaction is a transfer of ownership in response to an instruction, transfer of ownership documentation is placed in either the first trusted third-party repository of information objects or the second trusted third-party repository of information objects, the trusted third-party repository of information objects having the transfer of ownership documentation validates authenticity of the transfer of ownership documentation by verifying all digital signatures, certificate validity periods, and using the CSS to check certificate status of all authentication certificates included in the transfer of ownership documentation, appends a date and time indication, and digitally signs, wraps and stores the transfer of ownership documentation, which are added to the manifest.

32. (Withdrawn) The method of claim 19, wherein certificate status is indicated to the CSS by a Certificate Revocation List ("CRL"), according to a publication schedule of the issuing CA, the CSS retrieves the CRL from a certificate status reporting component listed in the configuration store, the

CSS clears a cache memory associated with the issuing CA, and the CSS determines the status of the authentication certificate from the CRL and stores the status in the cache memory associated with the issuing CA.

33. (Withdrawn) The method of claim 19, wherein certificate status is indicated to the CSS by a Delta Certificate Revocation List ("ΔCRL"); upon notification by the issuing CA that a ΔCRL is available, the CSS retrieves the ΔCRL from a certificate status reporting component listed in the configuration store; if the ΔCRL is a complete CRL, then the CSS clears a cache memory associated with the issuing CA, determines the status from the CRL, and stores the status in the cache memory; and if the ΔCRL contains only changes occurring after publication of a full CRL, the CSS determines the status from the ΔCRL, and stores the status in the cache memory.

34. (Currently Amended) The method of claim [[5]] 18, wherein a ~~background low priority garbage collection utility~~ cleanup process removes all stale cache entries as required when new CRLs or ΔCRLs are retrieved, one of the thresholds is exceeded, or freeing up of cache is required. ~~where the time-to-live data element exceeds current local time and/or may initiate a status update if established is CSS policy.~~

35. (Currently Amended) The method of claim 1, ~~whereby~~ wherein any [[one]] CSS[[,]] can query any other CSS for the certificate status if that CSS is designated in the configuration store as an approved certificate status

reporting component for the issuing CA. ~~primary, retrieves certificate status from a CA, PKI, or certificate status server and any other CSS, designated secondary, queries the primary CSS for certificate status.~~